Machine Learning Optimization Summary

No Author Given

No Institute Given

Abstract. **Keywords:** machine learning optimization, newton's method, gradient descent, fminsearch

1 Unconstrained Optimization Methods

1.1 Gradient Descent

Gradient Descent (GD) method is derivative related method. If we have a function $f(\cdot)$ and its gradient is $\bigtriangledown f(\cdot)$.

$$\bigtriangledown f(\cdot) = \frac{\partial f(\cdot)}{x}\overline{x} + \frac{\partial f(\cdot)}{y}\overline{y} + \frac{\partial f(\cdot)}{z}\overline{z} + \cdots \tag{1}$$

Note that the gradient is a vector. The number of terms in Eq. (1) depends how many directions the function $f(\cdot)$ has. For example, $f(x, y) = x^2 + 3y^2$ has two directions: $\overline{x}$ and $\overline{y}$. Its gradient at point (x, y) is:

$$\bigtriangledown f(x, y) = 2x\overline{x} + 6y\overline{y} \tag{2}$$

To determine the next point in the descending procedure, we need a function as follows:

$$\xi(\tau) = \overline{x} - \tau \cdot \bigtriangledown f(\overline{x}), \tag{3}$$

where $\overline{x}$ is the current location. This equation is telling the next movement is going τ amount against (minus) the gradient of the current location. Also, $-\bigtriangledown f(\cdot)$ is the direction of steepest descent of f at (x, y) How to calculate the parameter τ? Actually, we can take Eq. (3) back to function $f(\cdot)$ to represent the next targeting point: $f(\xi(\tau))$. Note that as we aim to move τ amount against the gradient direction at $\overline{x}$ achieving the right point $f(\xi(\tau))$ where the gradient would be the minimal (=0). τ would be the optimal for that movement. Thus, we can get the derivative of the new point with respect to τ and set it to zero to obtain the optimal τ.

$$\frac{df(\xi(\tau))}{d\tau} = -\tau \bigtriangledown f(\xi(\tau)) \cdot \bigtriangledown f(\overline{x}) = -\tau \cdot \bigtriangledown f(\overline{x\prime}) \cdot f(\overline{x}) = 0 \tag{4}$$

Note that $\overline{x\prime}$ represents the new point after the movement. From this equation, we can see that the optimal τ can make the gradient at the new point $(\bigtriangledown f(\overline{x\prime}))$ orthogonal to the gradient at the current location $(\bigtriangledown f(\overline{x})$ as inner product of their gradients is equal to 0. Let's go back to the example function $f(x, y) =$

$x^2 + 3y^2$. In Eq. (2), we have its gradient at (x, y). The function $f(\xi(\tau))$ should be:

$$f(\xi(\tau)) = (1 - 2\tau)^2 x^2 + 3(1 - 6\tau)^2 y^2 \tag{5}$$

Then its derivative with respect to τ is

$$\begin{aligned} f'(\xi(\tau)) = \frac{df(\xi(\tau))}{d\tau} = 2(1 - 2\tau)(-2)x^2 + 3 \cdot 2(1 - 6\tau)(-6)y^2 = 0 \\ \Rightarrow x^2 - 2\tau x^2 + 9y^2 - 54\tau y^2 = 0 \\ \Rightarrow \tau = \frac{x^2 + 9y^2}{2x^2 + 54y^2} \end{aligned} \tag{6}$$

Note that we can indeed get the optimal τ for each movement, however, the cost if very high if we have high dimensional data. According to Eq. (3), we can have $\xi = (1-2*\tau)x\hat{x}+(1-6*\tau)y\hat{y}$. Then we take the optimal τ back to this equation to update the new x' and y' iteratively until the function converges.

In machine learning, we can use gradient descent to optimize objective function of linear regression. The objective function is:

$$f(\theta) = \frac{1}{n}(\boldsymbol{y} - \boldsymbol{X}\theta)^T(\boldsymbol{y} - \boldsymbol{X}\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \theta^T x_i)^2, \tag{7}$$

where $\boldsymbol{y} \in \mathbb{R}^{n\times 1}$, $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, $\theta \in \mathbb{R}^{d\times 1}$. n is the number of data sample while d is the number of feature dimension. Note that the loss function in linear regression is a squared loss (quadratic function). To use gradient descent to optimize (minimize) the loss, we can use Eq. (3) after we get the gradient of $f(\theta)$:

$$\bigtriangledown f(\theta) = -2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\theta = -\frac{2}{n}\sum_{i=1}^{n} x_i(y_i - \theta^T x_i) \tag{8}$$

Note that we are optimizing $f(\theta)$. Then we can have the gradient descent equation as follows:

$$\begin{aligned} \theta_{k+1} &= \theta_k - \eta \bigtriangledown f(\theta_k) \\ \Rightarrow \theta_{k+1} &= \theta_k - \eta(-2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\theta_k) \\ \Rightarrow \theta_{k+1} &= \theta_k - \eta[-\frac{2}{n}\sum_{i=1}^{n} x_i(y_i - \theta^T x_i)] \end{aligned} \tag{9}$$

As mentioned before, we can calculate the optimal η. However the cost is very high. So we just empirically tune the values of η. If η is too small, the convergence process will be extremely slow as each step is very small. In contrast, if η is very big, the objective function value will not converge.

1.2 Newton's Method

Newton's method is inspired by taylor expansion and is a second-order optimization method. Intuitively, we use Hessian in each iterative step when calculating θ.

$$\theta_{k+1} = \theta_k - \boldsymbol{H} \bigtriangledown f(\theta_k) \tag{10}$$

Hessian represents the curvature of function. In fact, Newton's method is inspired by Taylor expansion. The Taylor expansion is an approximation method.

$$f(x+\epsilon) = f(x) + \frac{1}{1!}f'(x)\epsilon + \frac{1}{2!}f''(x)\epsilon^2 + \cdots \tag{11}$$

Taylor expansion approximates a point $(x+\epsilon)$ close to x by using a Taylor series. Just keep first and second order of Taylor expansion above (higher order derivatives of quadratic equation are equal to zero), then we have:

$$f(\theta_{k+1}) = f(\theta_k) + f'(\theta_k)(\theta_{k+1}-\theta_k) + \frac{1}{2}f''(\theta_k)(\theta_{k+1}-\theta_k)^2, \tag{12}$$

We are aiming to achieve the optimal location at k+1-th step, so we set derivative of $f(\theta_{k+1}$ w.r.t. θ_{k+1} be zero:

$$\begin{aligned} f'(\theta_{k+1}) &= f'(\theta_k)f''(\theta_k)(\theta_{k+1}-\theta_k) \\ \theta_{k+1} &= \theta_k - \end{aligned} \tag{13}$$

1.3 fminsearch